MnM_Sampling_Notebook

Rutendo Sigauke

2025-07-11

Summary of R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

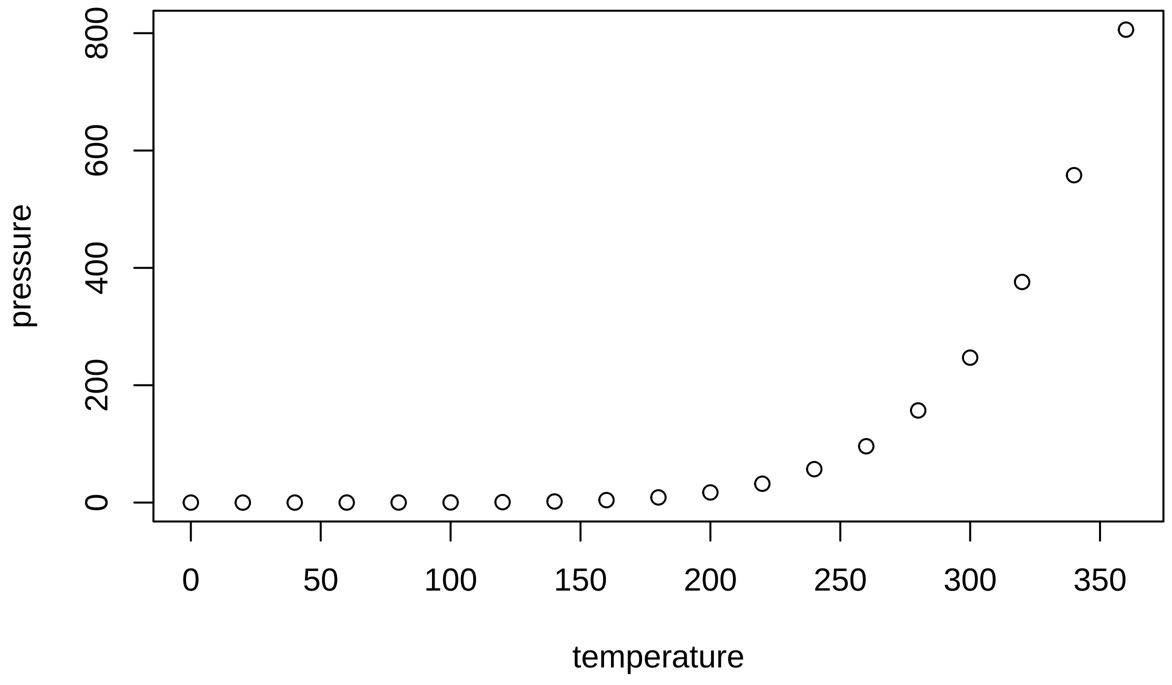
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```r
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00  
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

M&M Counting Exercise

Here we will use M&M counts from a previous Short Read workshop. In this experiment, we have two bowls (red and green) with M&Ms. The two bowls represent an experimental set up, say green is our control while red is our treated.

The goal of this exercise is to visualize how we count reads in Short Read sequencing and ways to treat bias (i.e. contamination).

Since this will be virtual, we will ask you to run the code cells below and explore the outputs.

Loading libraries

We will be using these libraries below. If you do not have them installed, please go ahead and do so.

```r
# First, install the required packages
#install.packages('data.table')
#install.packages('tidyverse)
#install.packages('ggplot2')
#install.packages('cowplot)

# Load the required library
library(data.table)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::between()     masks data.table::between()
## x dplyr::filter()      masks stats::filter()
## x dplyr::first()       masks data.table::first()
## x lubridate::hour()    masks data.table::hour()
## x lubridate::isoweek() masks data.table::isoweek()
## x dplyr::lag()         masks stats::lag()
## x dplyr::last()        masks data.table::last()
## x lubridate::mday()    masks data.table::mday()
## x lubridate::minute()  masks data.table::minute()
## x lubridate::month()   masks data.table::month()
## x lubridate::quarter() masks data.table::quarter()
## x lubridate::second()  masks data.table::second()
## x purrr::transpose()   masks data.table::transpose()
## x lubridate::wday()    masks data.table::wday()
## x lubridate::week()    masks data.table::week()
## x lubridate::yday()    masks data.table::yday()
## x lubridate::year()    masks data.table::year()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(ggplot2)
library(cowplot)
```

```
##
## Attaching package: 'cowplot'
##
## The following object is masked from 'package:lubridate':
##
##     stamp
```

Loading data

This are M&M counts from 2024. Note the summary statistics for each color of M&Ms. What other candies do you observe in out two bowls?

```
# Load the data from 2024 m&m counts as a data table
mnm_dt <- data.table::fread("~/Desktop/SR2025/day7/notebooks/MandM_all_2024.tsv")
dim(mnm_dt) #get the dimensions of datatable
```

```
## [1] 47 15
```

```
head(mnm_dt) # getting the top lines
```

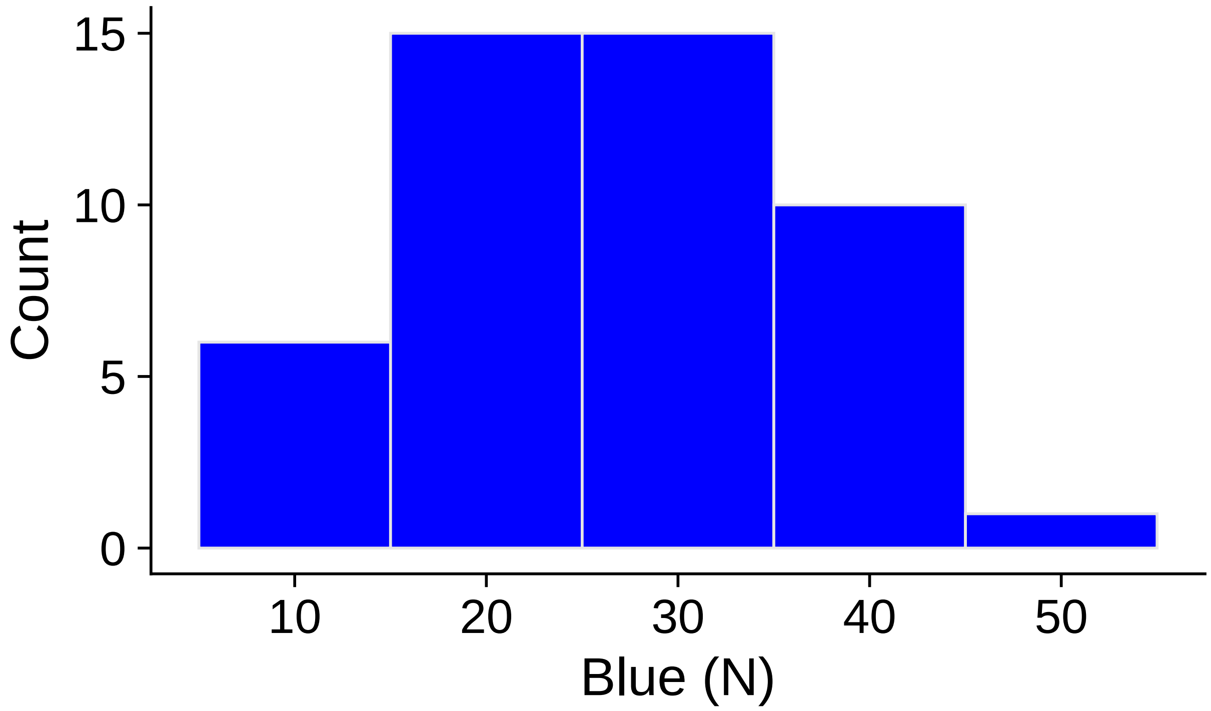
```
##      Name big marsh tiny marsh  blue brown orange yellow   red green
##    <char>     <int>      <int> <int> <int>  <int>  <int> <int> <int>
## 1:   Mary         5         13    36    14     22      9    17     6
## 2:  Robin         1         23    33     5      8      6    16    19
## 3:   Dave         3         24    20    12     17     10    14    21
## 4: Andrea         5         13    36    14     22      9    17     6
## 5:   Thea         2         28    32     2     22      6    10    11
## 6:  Megan         3         28    28     9     16     11    13    24
##    orange_sktl green_sktl red_sktl yellow_sktl purple_sktl
##          <int>      <int>    <int>       <int>       <int>
## 1:           0          0        0           0           0
## 2:           0          0        0           0           0
## 3:           0          0        0           0           0
## 4:           0          0        0           0           0
## 5:           0          0        0           0           0
## 6:           2          0        0           0           0
##    Green bowl or Red bowl
##                    <char>
## 1:                    red
## 2:                    red
## 3:                  green
## 4:                    red
## 5:                    red
## 6:                  green
```

```
# Rename columns and remove spaces, since it easier to work with
setnames(mnm_dt,
         c("big marsh", "tiny marsh","Green bowl or Red bowl"),
         c("big_marsh", "tiny_marsh","bowl"))
colnames(mnm_dt) #print the column names
```

```
##  [1] "Name"        "big_marsh"   "tiny_marsh"  "blue"        "brown"
##  [6] "orange"      "yellow"      "red"         "green"       "orange_sktl"
## [11] "green_sktl"  "red_sktl"    "yellow_sktl" "purple_sktl" "bowl"
```

Exploring the data

```
# Histogram of each color, binwidth will depend on approx size of samples
ggplot(mnm_dt, aes(x=blue)) +
  geom_histogram(binwidth=10, fill="blue", color="gray90") +
  labs(title = " ",
       y = "Count",
       x = "Blue (N)") +
  theme_cowplot(20) +
  theme(plot.title = element_text(hjust = 0.5),
        title = element_text(size = 30),
        axis.title = element_text(size = 20),
        axis.text.y = element_text(size = 18),
        axis.text.x = element_text(size = 18))
```



CHALLENGE: Plot the distribution of the other M&Ms

We can do this one M&M, skittle, marshmallow at a time. But, we can make this step simpler. In order to plot all the items in the data.table, we have to rearrange the the table from a **wide** for to a **long** form. There are a couple ways we can achieve this (1.) with `pivot_longer` from tidyr or (2) `melt` from reshape2.

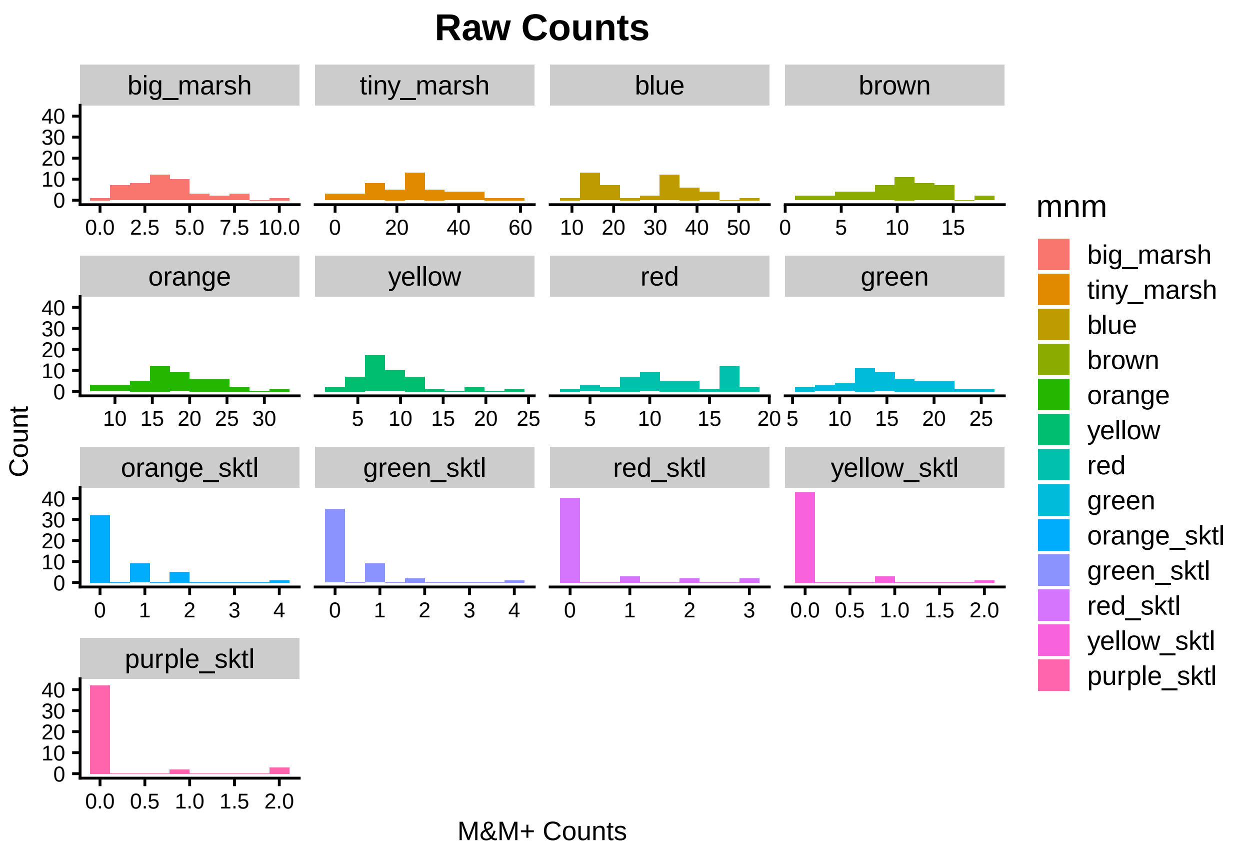
Below, I am using the `melt` function. Additional resources for reshaping a `data.table` can be found here.

```
# get long format for the table
mnm_long_dt <- melt(mnm_dt, id.vars = c("Name","bowl"),
                    measure.vars =c('big_marsh', 'tiny_marsh', 'blue',
                                    'brown', 'orange', 'yellow', 'red',
                                    'green',  'orange_sktl',
                                    'green_sktl', 'red_sktl',
                                    'yellow_sktl', 'purple_sktl'))
```

CHALLENGE: Compare the dimensions of the `mnm_dt` and the `mnm_long_dt`. How many row and columns do each of teh data.tables have? Feel free to also view the top 5 rows of each data.table.

```r
# If you have completed the CHALLENGE above, you will notice
#  the headings are generic. We can rename each of the columns below
colnames(mnm_long_dt) <- c("Name","bowl","mnm","count")
```

```r
ggplot(mnm_long_dt, aes(count, fill=mnm)) +
  geom_histogram(bins=10) +
  facet_wrap(~mnm, scales = 'free_x') +
  ggtitle("Raw Counts") +
  labs(title = "Raw Counts",
       y = "Count",
       x = "M&M+ Counts") +
  theme_cowplot(12) +
  theme(plot.title = element_text(hjust = 0.5),
        title = element_text(size = 12),
        axis.title = element_text(size = 10),
        axis.text.y = element_text(size = 8),
        axis.text.x = element_text(size = 8))
```



CHALLENGE: Keep only the M&M and plot the histogram of only the M&M (*blue*, *brown*, *orange*, *yellow*, *red*, *green*). Hint: You can use the `subset` function.

Comparing Green and Red Bowl Counts

Split the data into respective bowls

Here, to compare between the green and red bowl, we will split the M&M counts between the two bowls.

```
# Here we are splitting the bowls with M&M
mnm_redbowl_dt <- subset(mnm_dt, bowl == "red")
mnm_greenbowl_dt <- subset(mnm_dt, bowl == "green")

#summary of dimensions
dim(mnm_redbowl_dt)
```

```
## [1] 25 15
```

```
dim(mnm_greenbowl_dt)
```

```
## [1] 22 15
```

Calculate the Fold Change on Raw Counts

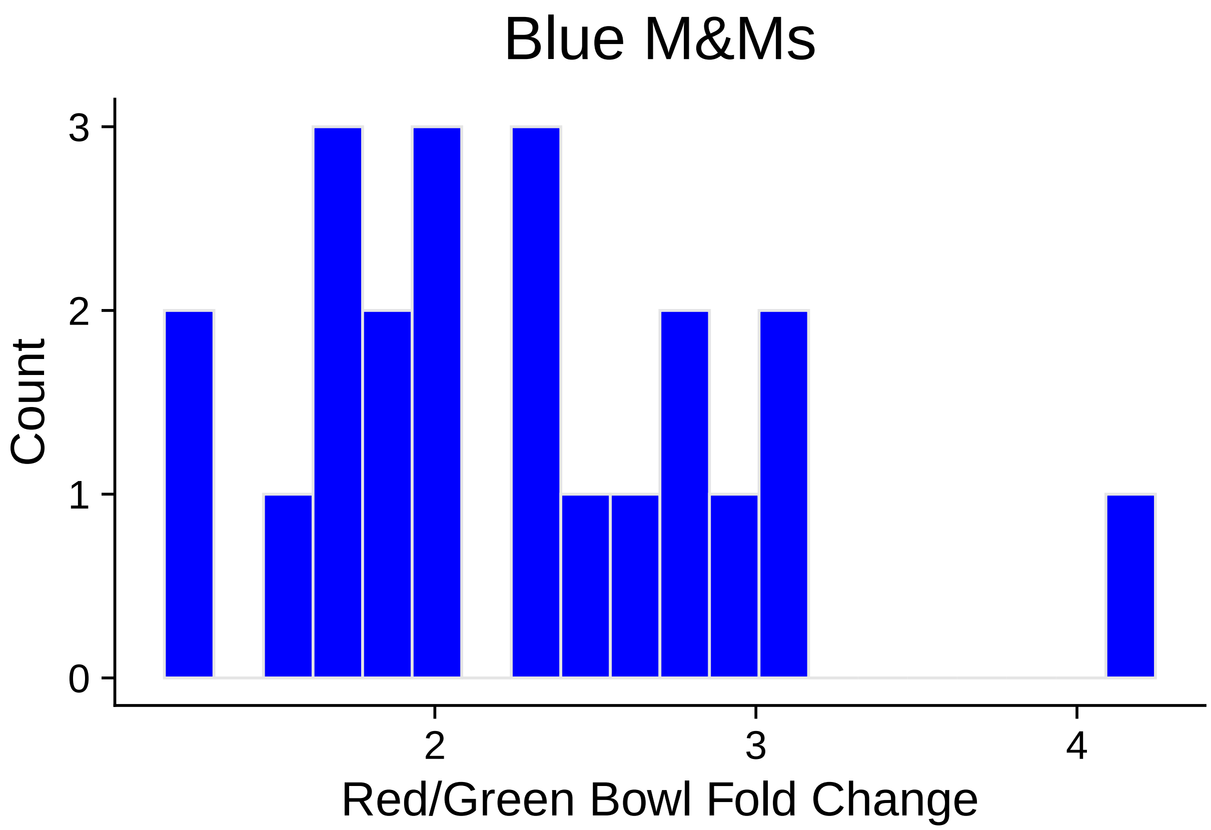
Note: Since we have uneven number of samples/individuals between the red and green bowl, we will have to use the smallest sample sizes for both data.tables to calculate the fold change

```
# 1: get value columns
mnm_redbowl_val_dt <- mnm_redbowl_dt[1:22,c('blue','brown','orange',
                                            'yellow','red','green')]

mnm_greenbowl_val_dt <- mnm_greenbowl_dt[1:22,c('blue','brown','orange',
                                                'yellow', 'red', 'green')]

# 2: calculate the fold change of the columns
mnm_redgreen_fc_dt <- mnm_redbowl_val_dt/mnm_greenbowl_val_dt
head(mnm_redgreen_fc_dt)
```

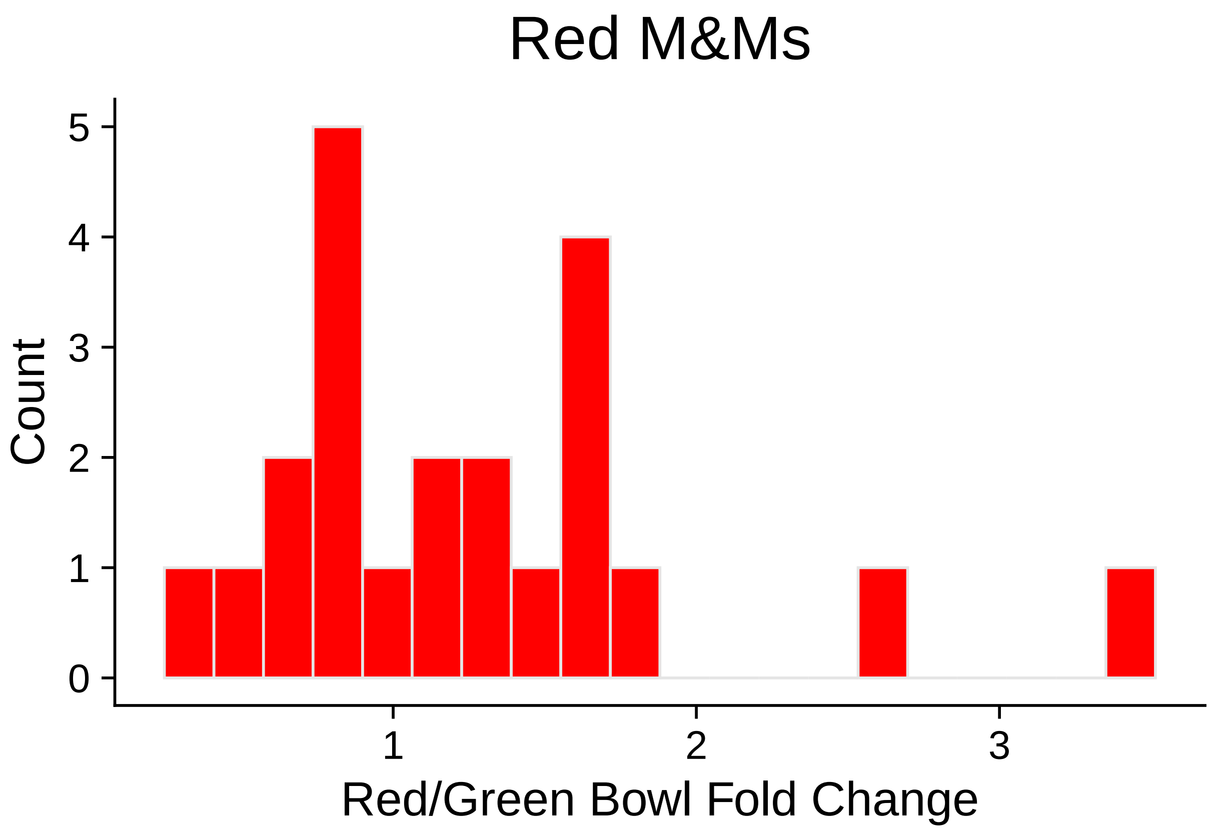
```
##        blue     brown    orange    yellow       red     green
##       <num>     <num>     <num>     <num>     <num>     <num>
## 1: 1.800000 1.1666667 1.2941176 0.9000000 1.2142857 0.2857143
## 2: 1.178571 0.5555556 0.5000000 0.5454545 1.2307692 0.7916667
## 3: 2.250000 1.5555556 2.2000000 0.4736842 3.4000000 0.3529412
## 4: 2.461538 0.1538462 0.9166667 0.2608696 0.8333333 0.6470588
## 5: 2.312500 0.3636364 0.7407407 0.5000000 0.6250000 1.1250000
## 6: 4.111111 1.4444444 1.7692308 0.6250000 1.6000000 0.4705882
```

We can plot histograms of the fold change calculations.

```
# Histogram of each color, binwidth will depend on approx size of samples
# we can pick number of bins instead of bin width
blue_fc_hist <- ggplot(mnm_redgreen_fc_dt, aes(x=blue)) +
                  geom_histogram(bins = 20, fill="blue", color="gray90") +
                  labs(title = "Blue M&Ms",
                       y = "Count",
                       x = "Red/Green Bowl Fold Change") +
                  theme_cowplot(20) +
                  theme(plot.title = element_text(hjust = 0.5, face="plain"),
                        title = element_text(size = 20),
                        axis.title = element_text(size = 18),
                        axis.text.y = element_text(size = 15),
                        axis.text.x = element_text(size = 15))

blue_fc_hist
```

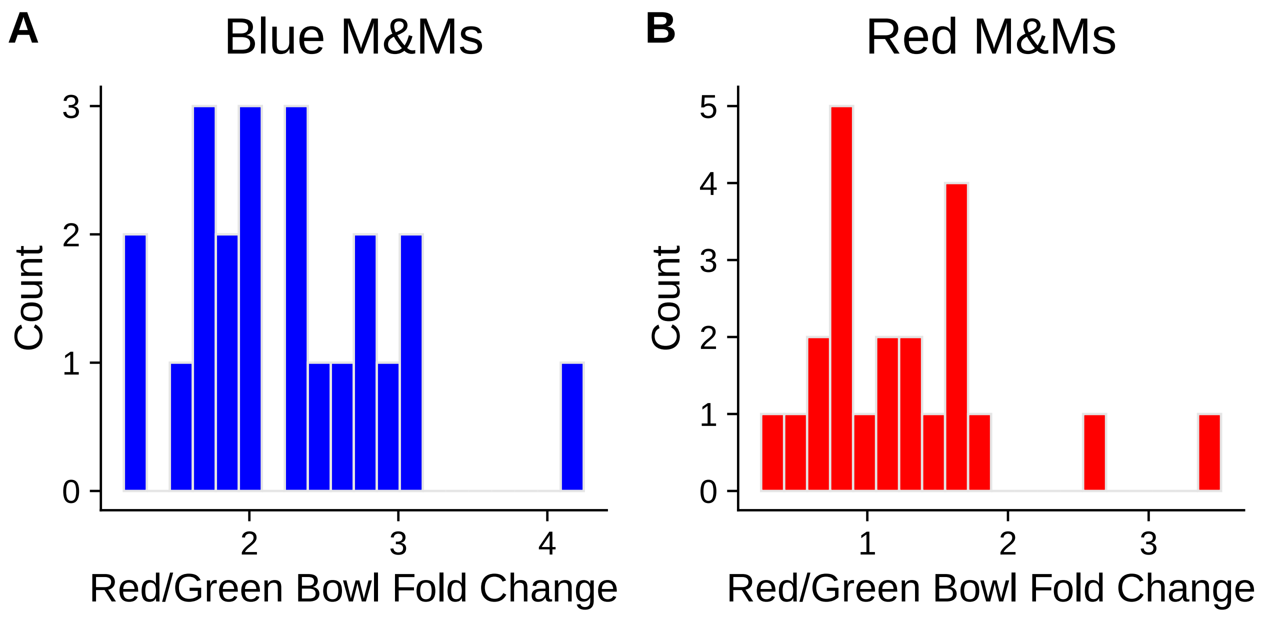


```
# Histogram of each color, binwidth will depend on approx size of samples
# we can pick number of bins instead of bin width
red_fc_hist <- ggplot(mnm_redgreen_fc_dt, aes(x=red)) +
                  geom_histogram(bins = 20, fill="red", color="gray90") +
                  labs(title = "Red M&Ms",
                       y = "Count",
                       x = "Red/Green Bowl Fold Change") +
                  theme_cowplot(20) +
                  theme(plot.title = element_text(hjust = 0.5, face="plain"),
                        title = element_text(size = 20),
                        axis.title = element_text(size = 18),
                        axis.text.y = element_text(size = 15),
                        axis.text.x = element_text(size = 15))

red_fc_hist
```



We can make publication ready plots using `cowplot`. See the cowplot vignette.

```r
red_blue_fc_hist  <- plot_grid(blue_fc_hist,
                              red_fc_hist,
                              labels = c('A', 'B'),
                              label_size = 20)
red_blue_fc_hist
```



We can save the plot using `ggsave` from ggplot2.

```
ggplot2::ggsave(filename = "~/Desktop/SR2025/day7/notebooks/red_blue_mnm_fc_hist.png",
       plot = red_blue_fc_hist,
       width = 8, height = 4)
```

Normalize the Counts by Sample Size

```
# 1: we will get row totals for each sample (individual)
mnm_redbowl_rowtotal <- rowSums(mnm_redbowl_val_dt)
mnm_greenbowl_rowtotal <- rowSums(mnm_greenbowl_val_dt)

# 2: divide each row item with the row total
mnm_redbowl_val_norm_dt <- mnm_redbowl_val_dt[,.SD/mnm_redbowl_rowtotal] # using data.table sytax
mnm_greenbowl_val_norm_dt <- mnm_greenbowl_val_dt[,.SD/mnm_greenbowl_rowtotal]

# 3: get the fold change for normalized counts
mnm_redgreen_norm_fc_dt <- mnm_redbowl_val_norm_dt/mnm_greenbowl_val_norm_dt
colnames(mnm_redgreen_norm_fc_dt) <- paste0(colnames(mnm_redgreen_norm_fc_dt),
                                            "_norm") #update the column names to show that they are nor
head(mnm_redgreen_norm_fc_dt)
```

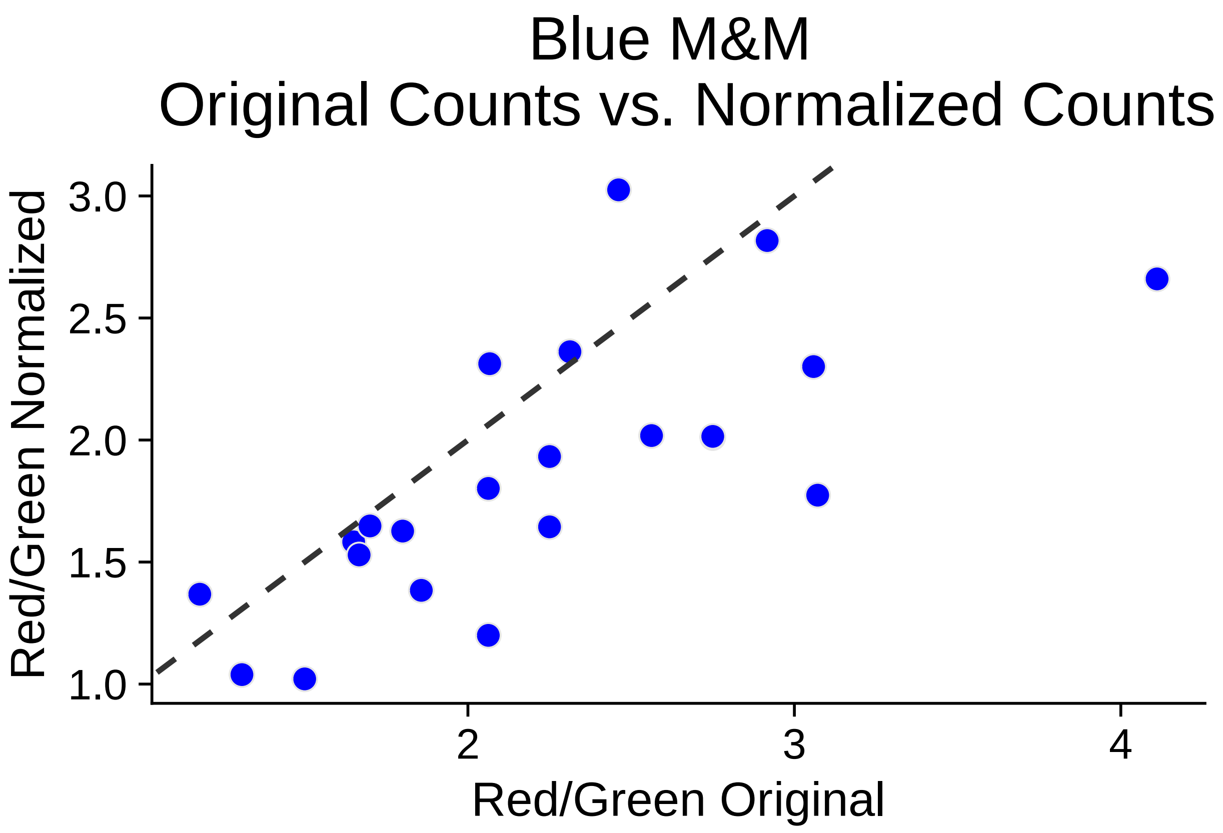
```
##    blue_norm brown_norm orange_norm yellow_norm  red_norm green_norm
##        <num>      <num>       <num>       <num>     <num>      <num>
## 1:  1.626923  1.0544872   1.1696833   0.8134615 1.0975275  0.2582418
## 2:  1.368227  0.6449553   0.5804598   0.6332288 1.4288240  0.9190613
## 3:  1.644231  1.1367521   1.6076923   0.3461538 2.4846154  0.2579186
## 4:  3.025023  0.1890639   1.1265060   0.3205867 1.0240964  0.7951807
## 5:  2.361702  0.3713733   0.7565012   0.5106383 0.6382979  1.1489362
## 6:  2.660131  0.9346405   1.1447964   0.4044118 1.0352941  0.3044983
```

Note: were are using `cbind` from base R. Best practices is to use an ID maintaining approaches like `merge` from base R or
`left_join` from dplyr.

```
# we can compare the raw count fold change with the normalized fold change.
mnm_redgreen_compare_fc_dt <- cbind(mnm_redgreen_fc_dt,
                                    mnm_redgreen_norm_fc_dt)
head(mnm_redgreen_compare_fc_dt)
```

```
##        blue     brown    orange    yellow       red     green blue_norm
##       <num>     <num>     <num>     <num>     <num>     <num>     <num>
## 1: 1.800000 1.1666667 1.2941176 0.9000000 1.2142857 0.2857143  1.626923
## 2: 1.178571 0.5555556 0.5000000 0.5454545 1.2307692 0.7916667  1.368227
## 3: 2.250000 1.5555556 2.2000000 0.4736842 3.4000000 0.3529412  1.644231
## 4: 2.461538 0.1538462 0.9166667 0.2608696 0.8333333 0.6470588  3.025023
## 5: 2.312500 0.3636364 0.7407407 0.5000000 0.6250000 1.1250000  2.361702
## 6: 4.111111 1.4444444 1.7692308 0.6250000 1.6000000 0.4705882  2.660131
##    brown_norm orange_norm yellow_norm  red_norm green_norm
##         <num>       <num>       <num>     <num>      <num>
## 1:  1.0544872   1.1696833   0.8134615 1.0975275  0.2582418
## 2:  0.6449553   0.5804598   0.6332288 1.4288240  0.9190613
## 3:  1.1367521   1.6076923   0.3461538 2.4846154  0.2579186
## 4:  0.1890639   1.1265060   0.3205867 1.0240964  0.7951807
## 5:  0.3713733   0.7565012   0.5106383 0.6382979  1.1489362
## 6:  0.9346405   1.1447964   0.4044118 1.0352941  0.3044983
```

```r
ggplot(mnm_redgreen_compare_fc_dt, aes(x=blue, y=blue_norm)) +
  geom_point(shape=21, fill="blue", color="gray90", size=4) +
  labs(title = "Blue M&M \n Original Counts vs. Normalized Counts",
       y = "Red/Green Normalized",
       x = "Red/Green Original") +
  geom_abline(intercept = 0, slope=1,
              linetype='dashed', color = 'gray20', linewidth=1) +
  theme_cowplot(20) +
  theme(plot.title = element_text(hjust = 0.5, face='plain'),
        title = element_text(size = 20),
        axis.title = element_text(size = 18),
        axis.text.y = element_text(size = 16),
        axis.text.x = element_text(size = 16))
```



Normalize the Counts by Marshmallows

Note, the number of marshmallows between the two bowls was kept constant.

```r
# 1: we will get marshmallow totals for each sample (individual)
mnm_redbowl_marshtotal <- rowSums(mnm_redbowl_dt[,c("big_marsh","tiny_marsh")])
mnm_greenbowl_marshtotal <- rowSums(mnm_greenbowl_dt[,c("big_marsh","tiny_marsh")])

# 2: divide each row item with the marshmallow total
mnm_redbowl_marsh_norm_dt <- mnm_redbowl_val_dt[,.SD/mnm_redbowl_marshtotal] # using data.table syntax
mnm_greenbowl_marsh_norm_dt <- mnm_greenbowl_val_dt[,.SD/mnm_greenbowl_marshtotal]

# 3: get the fold change for normalized counts
mnm_redgreen_marsh_norm_fc_dt <- mnm_redbowl_marsh_norm_dt/mnm_greenbowl_marsh_norm_dt
```

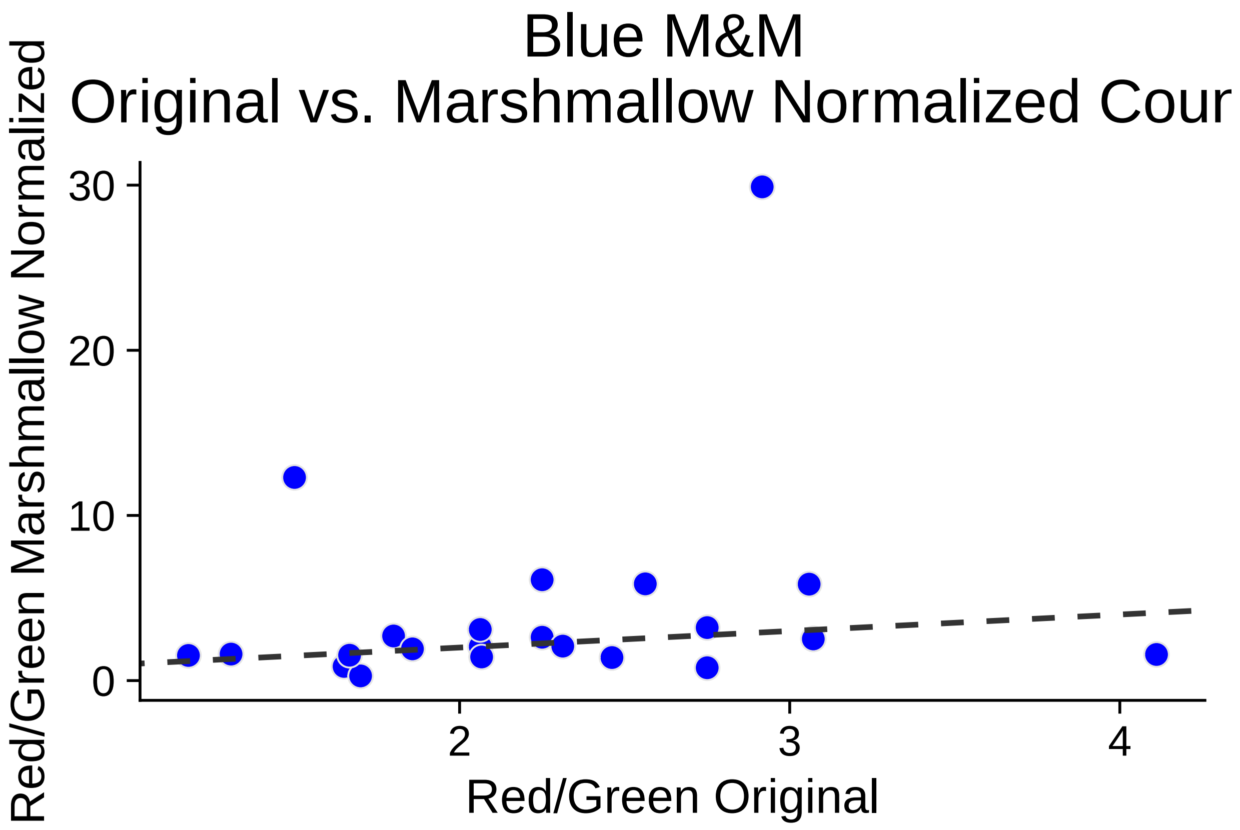
```
colnames(mnm_redgreen_marsh_norm_fc_dt) <- paste0(colnames(mnm_redgreen_marsh_norm_fc_dt),
                                          "_marshnorm") #update the column names to show that they ar
head(mnm_redgreen_marsh_norm_fc_dt)
```

```
##    blue_marshnorm brown_marshnorm orange_marshnorm yellow_marshnorm
##             <num>           <num>            <num>            <num>
## 1:       2.700000       2.1000000        6.9882353        0.6394737
## 2:       1.522321       0.3743961        0.3369565        4.2272727
## 3:       2.625000       0.8596491        2.5666667        0.4736842
## 4:       1.394872       0.1452991        1.0388889        0.8869565
## 5:       2.086890       0.5606061        0.5958132        0.4021739
## 6:       1.581197       0.8024691        0.4655870        0.3472222
##    red_marshnorm green_marshnorm
##            <num>           <num>
## 1:     0.9935065       0.1377551
## 2:     1.5897436       0.8462644
## 3:     3.2454545       0.1900452
## 4:     0.3728070       0.3333333
## 5:     5.7812500       1.7343750
## 6:     0.7619048       0.2139037
```

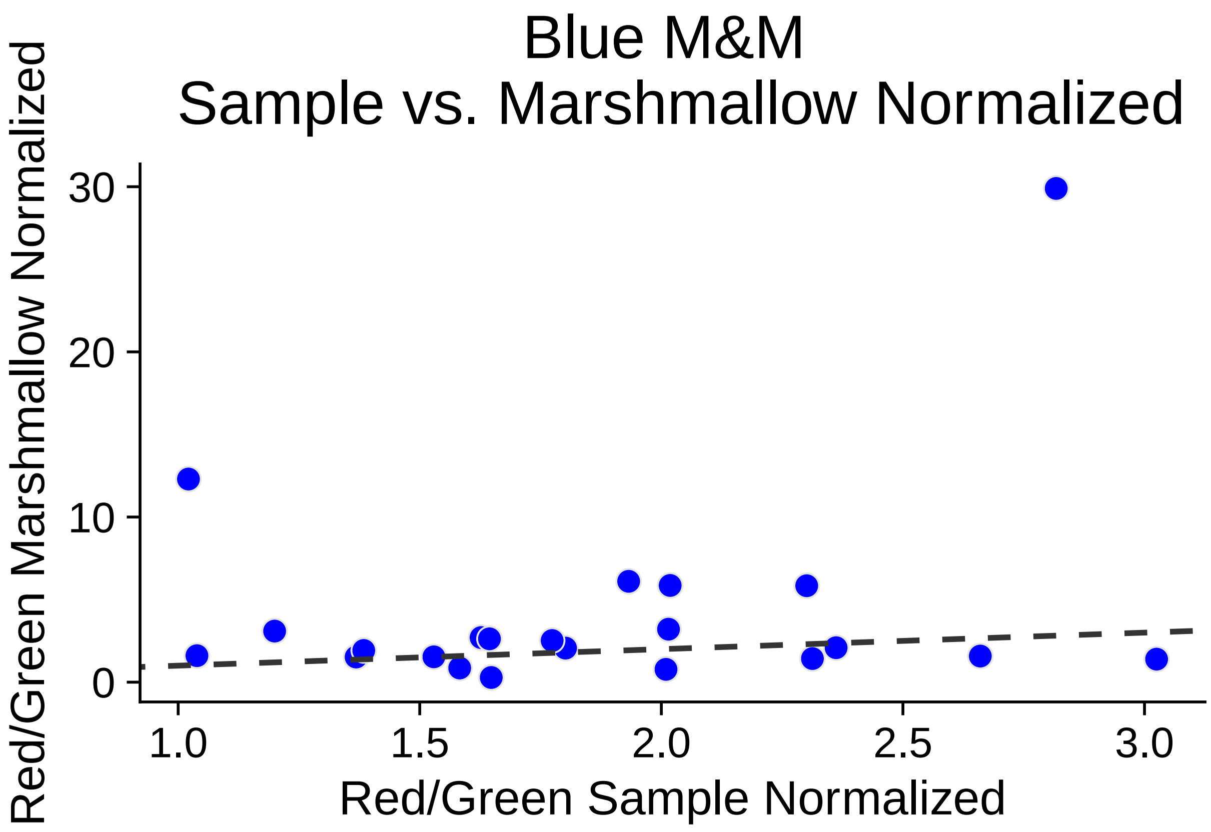
```
# we can compare the raw count fold change with the normalized fold change.
mnm_redgreen_compare_all_fc_dt <- cbind(mnm_redgreen_compare_fc_dt,
                                        mnm_redgreen_marsh_norm_fc_dt)
head(mnm_redgreen_compare_all_fc_dt)
```

```
##        blue     brown    orange    yellow       red     green blue_norm
##       <num>     <num>     <num>     <num>     <num>     <num>     <num>
## 1: 1.800000 1.1666667 1.2941176 0.9000000 1.2142857 0.2857143  1.626923
## 2: 1.178571 0.5555556 0.5000000 0.5454545 1.2307692 0.7916667  1.368227
## 3: 2.250000 1.5555556 2.2000000 0.4736842 3.4000000 0.3529412  1.644231
## 4: 2.461538 0.1538462 0.9166667 0.2608696 0.8333333 0.6470588  3.025023
## 5: 2.312500 0.3636364 0.7407407 0.5000000 0.6250000 1.1250000  2.361702
## 6: 4.111111 1.4444444 1.7692308 0.6250000 1.6000000 0.4705882  2.660131
##    brown_norm orange_norm yellow_norm  red_norm green_norm blue_marshnorm
##         <num>       <num>       <num>     <num>      <num>          <num>
## 1:  1.0544872   1.1696833   0.8134615 1.0975275  0.2582418       2.700000
## 2:  0.6449553   0.5804598   0.6332288 1.4288240  0.9190613       1.522321
## 3:  1.1367521   1.6076923   0.3461538 2.4846154  0.2579186       2.625000
## 4:  0.1890639   1.1265060   0.3205867 1.0240964  0.7951807       1.394872
## 5:  0.3713733   0.7565012   0.5106383 0.6382979  1.1489362       2.086890
## 6:  0.9346405   1.1447964   0.4044118 1.0352941  0.3044983       1.581197
##    brown_marshnorm orange_marshnorm yellow_marshnorm red_marshnorm
##              <num>            <num>            <num>         <num>
## 1:       2.1000000        6.9882353        0.6394737     0.9935065
## 2:       0.3743961        0.3369565        4.2272727     1.5897436
## 3:       0.8596491        2.5666667        0.4736842     3.2454545
## 4:       0.1452991        1.0388889        0.8869565     0.3728070
## 5:       0.5606061        0.5958132        0.4021739     5.7812500
## 6:       0.8024691        0.4655870        0.3472222     0.7619048
##    green_marshnorm
##              <num>
## 1:       0.1377551
## 2:       0.8462644
## 3:       0.1900452
## 4:       0.3333333
## 5:       1.7343750
## 6:       0.2139037
```

```
ggplot(mnm_redgreen_compare_all_fc_dt, aes(x=blue, y=blue_marshnorm)) +
  geom_point(shape=21, fill="blue", color="gray90", size=4) +
  labs(title = "Blue M&M \n Original vs. Marshmallow Normalized Counts",
       y = "Red/Green Marshmallow Normalized",
       x = "Red/Green Original") +
  geom_abline(intercept = 0, slope=1,
              linetype='dashed', color = 'gray20', linewidth=1) +
  theme_cowplot(20) +
  theme(plot.title = element_text(hjust = 0.5, face='plain'),
        title = element_text(size = 20),
        axis.title = element_text(size = 18),
        axis.text.y = element_text(size = 16),
        axis.text.x = element_text(size = 16))
```



```
ggplot(mnm_redgreen_compare_all_fc_dt, aes(x=blue_norm, y=blue_marshnorm)) +
  geom_point(shape=21, fill="blue", color="gray90", size=4) +
  labs(title = "Blue M&M \n Sample vs. Marshmallow Normalized",
       y = "Red/Green Marshmallow Normalized",
       x = "Red/Green Sample Normalized") +
  geom_abline(intercept = 0, slope=1,
              linetype='dashed', color = 'gray20', linewidth=1) +
  theme_cowplot(20) +
  theme(plot.title = element_text(hjust = 0.5, face='plain'),
        title = element_text(size = 20),
        axis.title = element_text(size = 18),
        axis.text.y = element_text(size = 16),
        axis.text.x = element_text(size = 16))
```



###TO DO 1: Fold change on all counts combined between bowls. 2: Compare those to the individual fold changes